Resolution of Large Scale Eigenproblem using YML Workflow Framework

Nahid Emad
Versailles university
France

Tetsuya Sakurai
Tsukuba university
Japan

Goals

Compute the Ritz-pairs $(\lambda_1,\ldots,\lambda_m)$, $(q_1,\ldots,q_m)$ of

$$\boxed{(A-\lambda B)q = 0 \text{ or } (A-\lambda I)\, q = 0}$$

by Sakurai-Sugiura and Padé-Rayleigh-Ritz (PRR) projection moment-based methods with

omniRPC middleware and YML workflow framework

Study of the problems:

- Algorithmic and implementation
- Performance evaluation

Outline

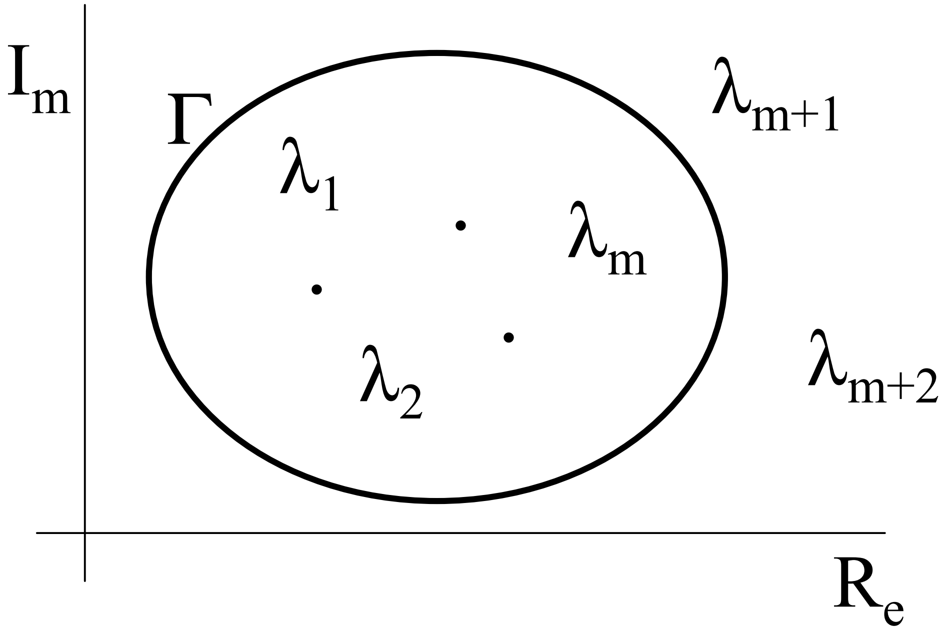
- Goals
- Sakurai-Sugiura method
- PRR method
- Distribution
  - ✓ Algorithmic
  - ✓ Implementation
- YML framework/omniRPC middleware
- Experiments
- Conclusion

Sakurai-Sugiura method

The generalized eigenvalue problem:

$$A\boldsymbol{x} = \lambda B\boldsymbol{x}$$

Suppose that m eigenvalues are located inside Γ domain



Moment-based method

$\boldsymbol{u}, \boldsymbol{v}$: nonzero vectors

Moments : $\mu_k = \dfrac{1}{2\pi\mathrm{i}} \displaystyle\int_\Gamma (z-\gamma)^k \boldsymbol{u}^H (zB - A)^{-1} \boldsymbol{v} dz$

Hankel Matrices :

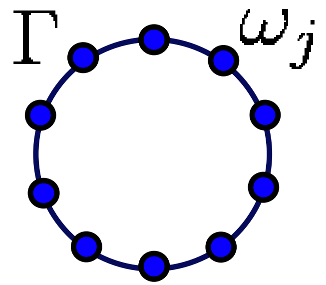
$$H_m = \begin{pmatrix} \mu_0 & \mu_1 & \cdots & \mu_{m-1} \\ \mu_1 & \mu_2 & \cdots & \mu_m \\ \vdots & \vdots & & \vdots \\ \mu_{m-1} & \mu_m & \cdots & \mu_{2m-2} \end{pmatrix} \quad H_m^< = \begin{pmatrix} \mu_1 & \mu_2 & \cdots & \mu_m \\ \mu_2 & \mu_3 & \cdots & \mu_{m+1} \\ \vdots & \vdots & & \vdots \\ \mu_m & \mu_m & \cdots & \mu_{2m-1} \end{pmatrix}$$

The eigenvalues of $H_m^< - \lambda H_m$ are given by $\lambda_1, \ldots, \lambda_m$

which are the poles of the Padé approximants of $\sum_{k=0,\infty} (\mu_{k}/z^{k+1})$

Circular domaine

Γ : Circle with center γ and radius ρ



Equidistributed points on the circle:

$$\omega_j = \gamma + \rho e^{\frac{2\pi \mathrm{i}}{N}(j+1/2)}, \quad j = 0, 1, \ldots, N-1$$

μ_k are approximated by the *N*-point trapezoidal rule:

$$\mu_k \approx \frac{1}{N} \sum_{j=0}^{N-1} (\omega_j - \gamma)^{k+1} f_j, \quad k = 0, \ldots, 2m-1$$

$$f_j = \boldsymbol{u}^H (\omega_j B - A)^{-1} \boldsymbol{v}, \quad j = 0, \ldots, N-1$$

Sakurai-Sugiura algorithm

input: u, v,N, m, γ, ρ ***output:*** $(\lambda_1,\ldots,\lambda_m)$, $(q_1,\ldots,q_m)$

1. Set $\omega_j=\gamma+\rho.\exp(2\pi i(j+1/2)/N)$, for $j = 0, \ldots N-1$.

2. $\mathbf{v=(\omega_j B-A)\ y_j}$**, for** $\mathbf{j = 0, \ldots N-1}$.

3. $f_j = u^H y_j$, for $j = 0, \ldots N-1$.

4. Compute μ_k, for $k = 0, \ldots 2m-1$.

5. Compute the eigenvalues $\xi_1, \ldots, \xi_m$ of $(H^<_m-\xi\hat{H}_m)z=0$.

6. Compute $q_1,\ldots,q_m$ by $q_j=W_m s_k$ where $s_k = \mu_k u$.

7. Set $\lambda_{j=}\gamma+\xi_j$, for $j = 0, \ldots, m-1$.

Padé-Rayleigh-Ritz method

X : non-zero vector

A: n-order Hermitian matrix

PRR approximates the poles of $R_x(\lambda) = ((I-\lambda A)^{-1}x, x)$

by those of $[m-1/m]_{Rx(\lambda)} = P_{m-1}(\lambda)/Q_m(\lambda)$

the Padé approximant of order m of the function $R_x(\lambda)$

The roots of $Q_m(\lambda)$ are given by $\lambda_1, \ldots, \lambda_m$

Moments : $\mu_{i+j}=(A^{i+j}x, x)$ for $i,j = 0, 1, \ldots$

Moment-based method

Let $H(i,j) = \begin{pmatrix} \mu_{i+0} & \mu_{i+1} & \cdots & \mu_{i+j} \\ \mu_{i+1} & \mu_{i+2} & \cdots & \mu_{i+j+1} \\ \vdots & \vdots & & \vdots \\ \mu_{i+j} & \mu_{i+j+1} & \cdots & \mu_{i+2j} \end{pmatrix}$ for $i,j = 0, 1, \cdots$

- Solve $H_m b = c$

where $H_m = H(0, m-1)$, $c = -(\mu_m, \mu_{m+1}, \ldots, \mu_{2m-1})^t$ and
$b = -(b_0, b_1, \ldots, b_{m-1})^t$

- Compute the roots of $Q_m(\lambda) = \lambda^m + b_{m-1}\lambda^{m-1} + \ldots + b_1\lambda + b_0$
whose the companion matrix is :

$$Q_m^c = \begin{pmatrix} -b_{m-1} & -b_{m-2} & \ldots & -b_1 & -b_0 \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & 0 \end{pmatrix}$$

PRR algorithm

input: x_0, k, m ***output:*** $(\lambda_1,\ldots,\lambda_m)$, $(q_1,\ldots,q_m)$

1. Compute $\mu_{2k-1}=(x_k, x_{k-1})$, $\mu_{2k}=(x_k, x_k)$
where $x_k = Ax_{k-1}$, for $k = 1, \ldots, m-1$.

2. Solve the linear system: $H_m b = c$.

3. Compute the eigenvalues λ_i of the matrix H_m

4. Compute the vectors z_i of $(H_m\text{-}\lambda H^{<}{}_m)z_i=0$

5. Compute the approximated eigenvectors
$[q_1,\ldots,q_m] = [x_1,\ldots,x_m][z_1,\ldots,z_m]$.

5. If no convergence then with a new x_0, go to 1.

Some properties of both approaches

Sakurai-Sugiura:

- solve generalized eigenproblem
- is not iterative
- acquires N n-order linear systems solving
- numerically reliable

Padé-Raleigh-Ritz:

- Solve Hermitian eigenproblem
- iterative
- acquires an m-order linear system solving
- numerically reliable for *small* subspaces

Multiple PRR

Multiple PRR

$$m_i \in [m_{min}, \ldots, m = m_{max}]$$

- Different subspaces
  - $K_{mi} = span\{w_i, Aw_i, \ldots, A^{mi-1}w_i\}$ *with*

    $m_i \neq m_j$ *and* $W_i \neq W_j$ *for* $i,j \in [1,\ldots,l]$ *and* $i \neq j$

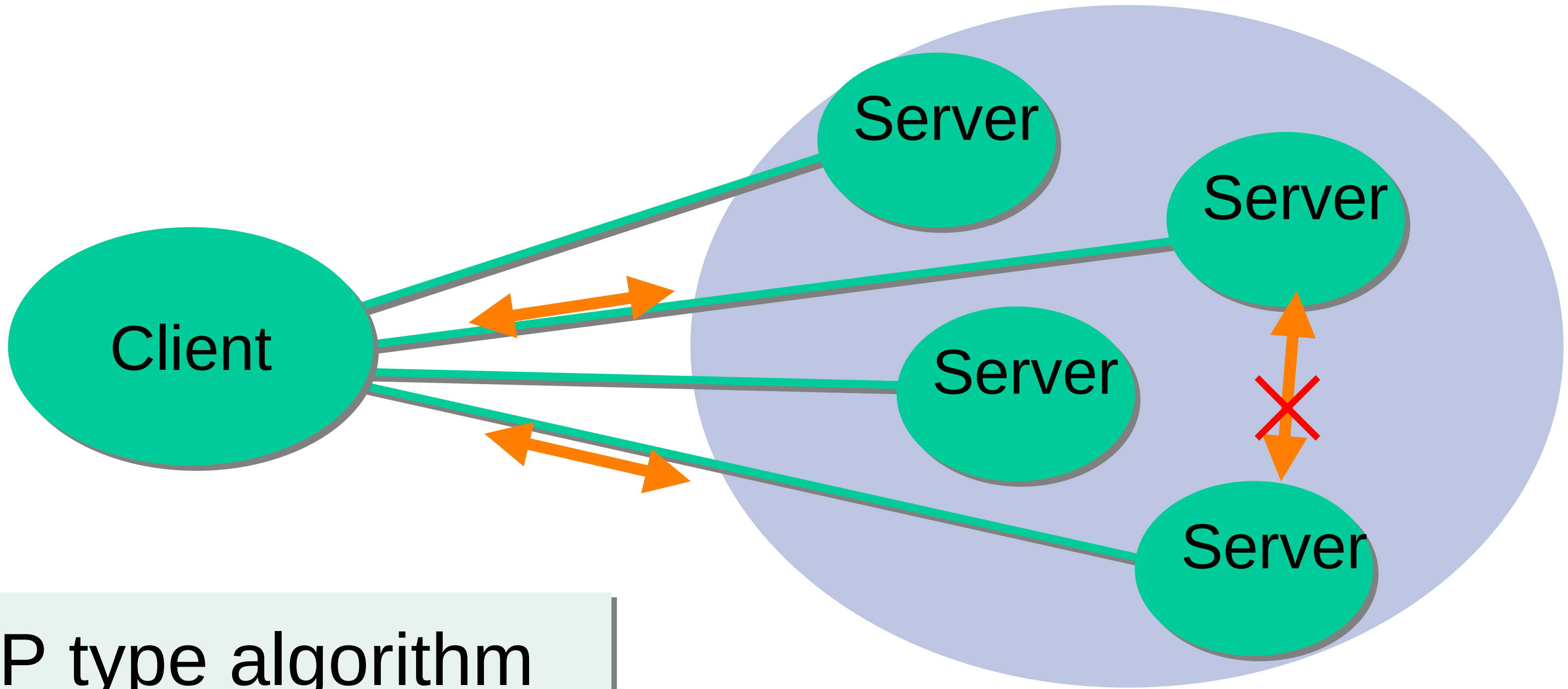
- « Nested » subspaces
  - $K_{ml} = span\{w_1, Aw_1, \ldots, A^{m1-1}w_1, A^{m1}W_2, \ldots, A^{m2-1}W_2, A^{m2}W_3, \ldots, A^{m3-1}W_3, \ldots, A^{ml-1}W_l\}$

    $K_{m1} \subset K_{m2} \subset \ldots \subset K_{ml}$

Computation with distributed resources

For a GRID environment:

- Master-worker type algorithm

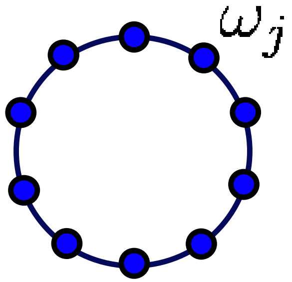


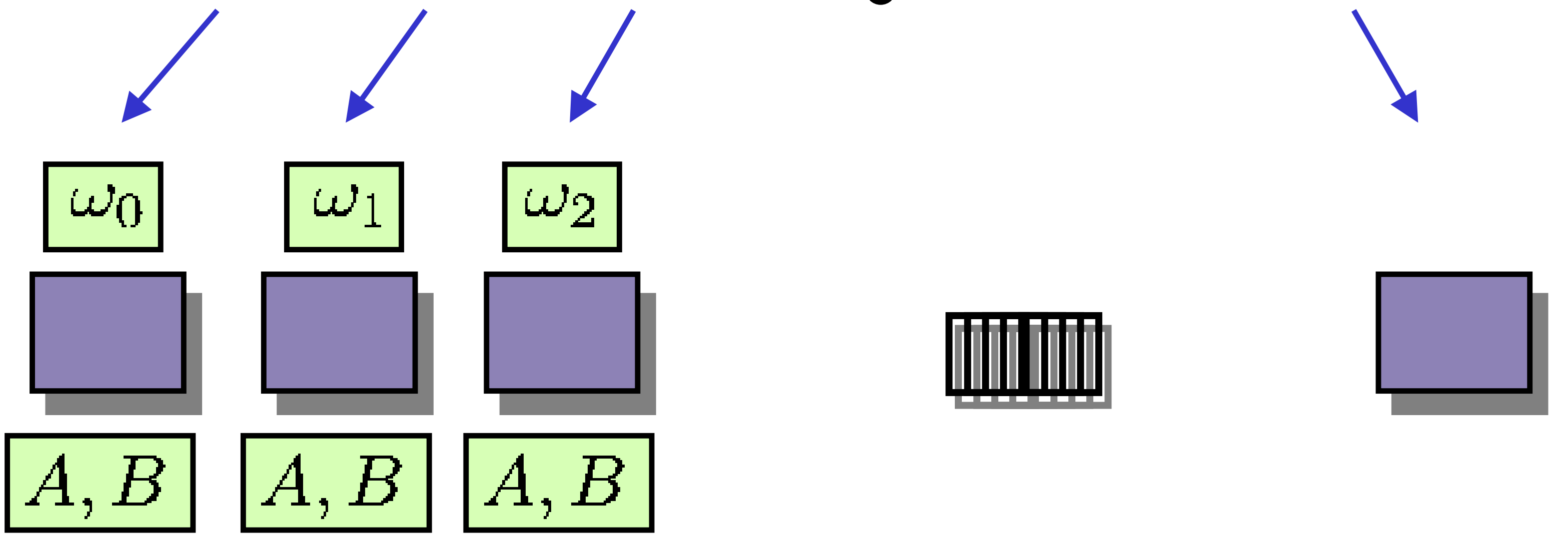
- P2P type algorithm

Sakurai-Sugiura method distribution

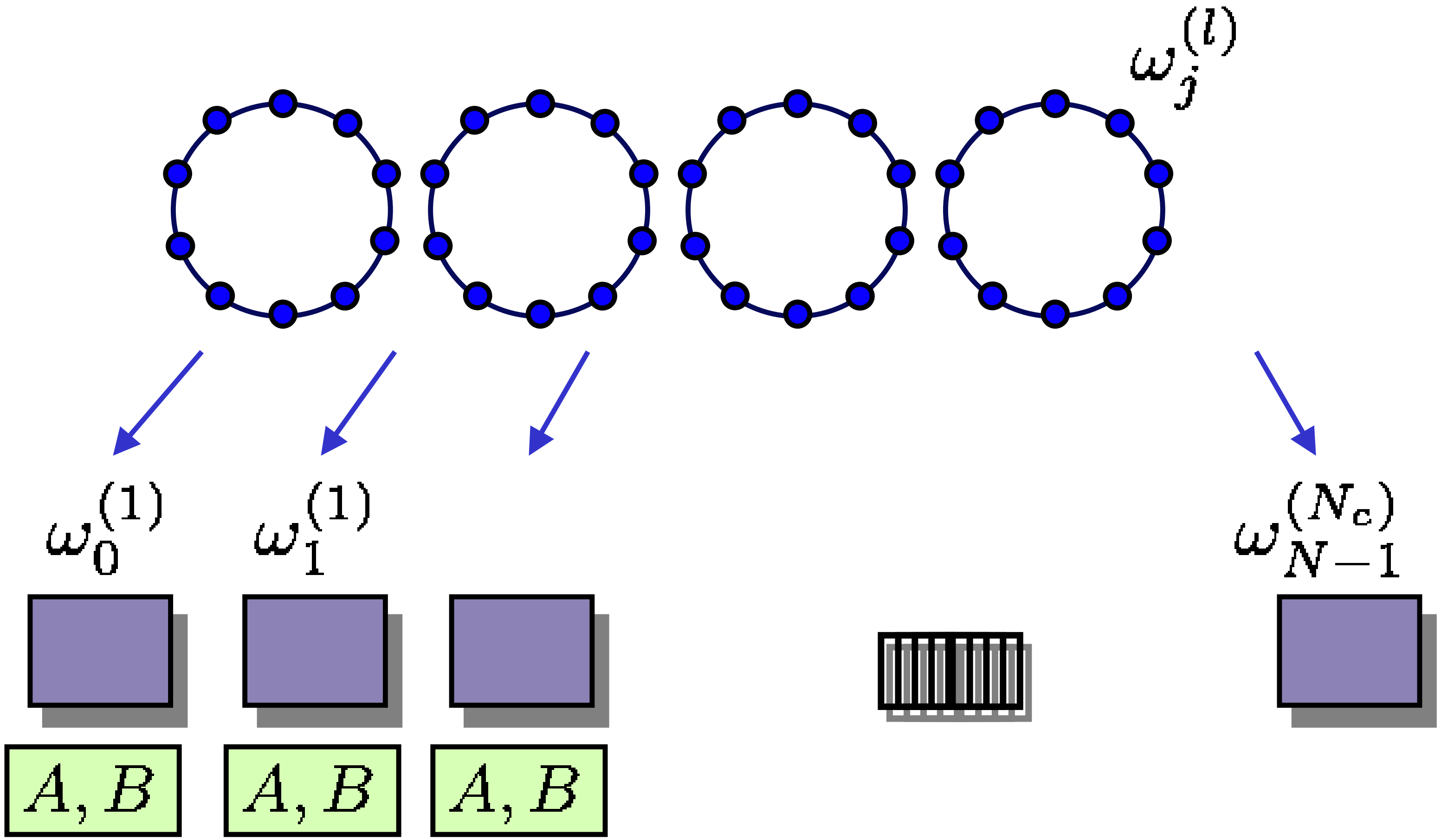
$$\mu_k \approx \frac{1}{N} \sum_{j=0}^{N-1} (\omega_j - \gamma)^{k+1} f_j$$

$$f_j = \boldsymbol{u}^H (\omega_j B - A)^{-1} \boldsymbol{v}$$





Sakurai-Sugiura method distribution



Multiple PRR distribution

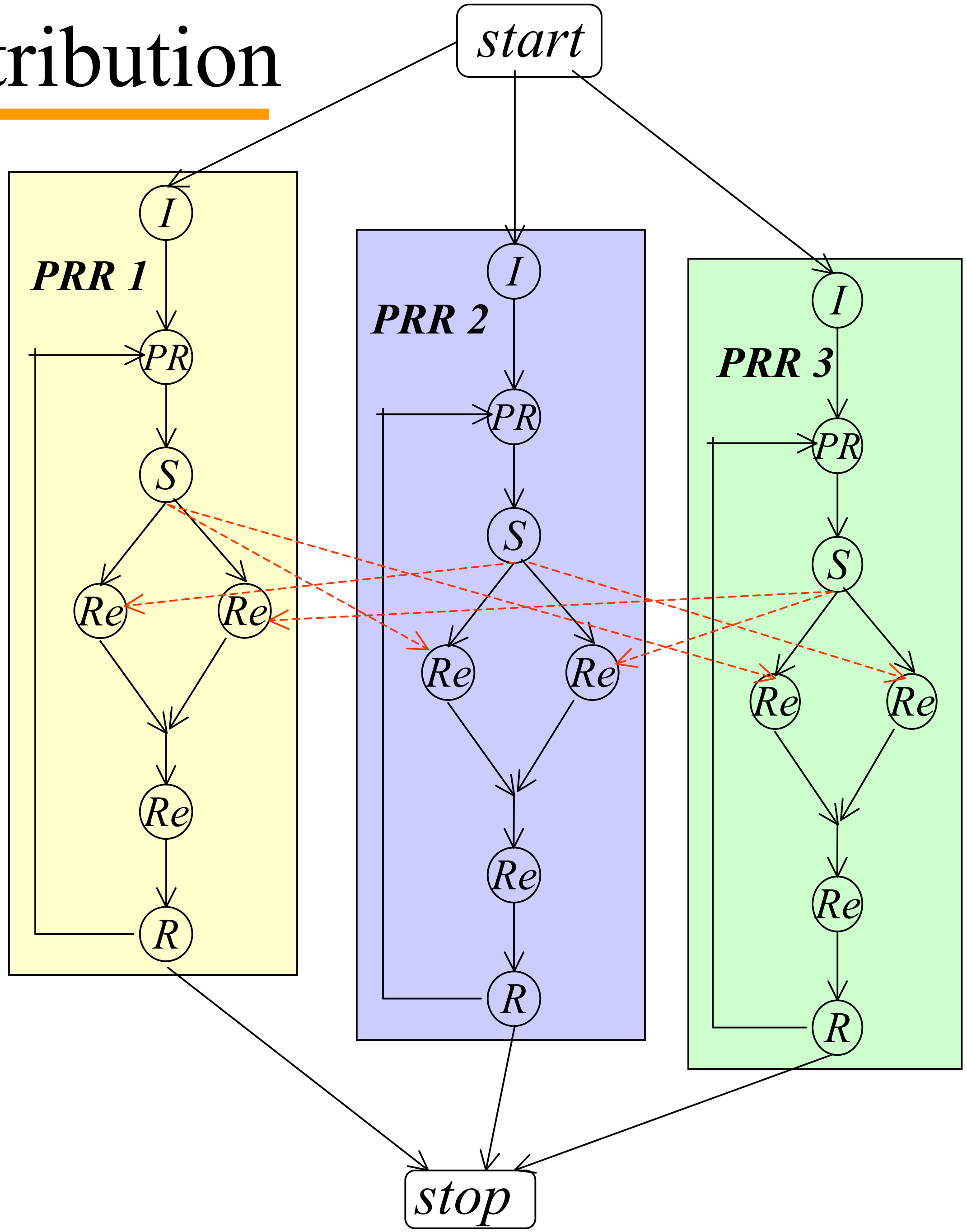
I : *Iinitialisation*

PR : *PRR reduction*

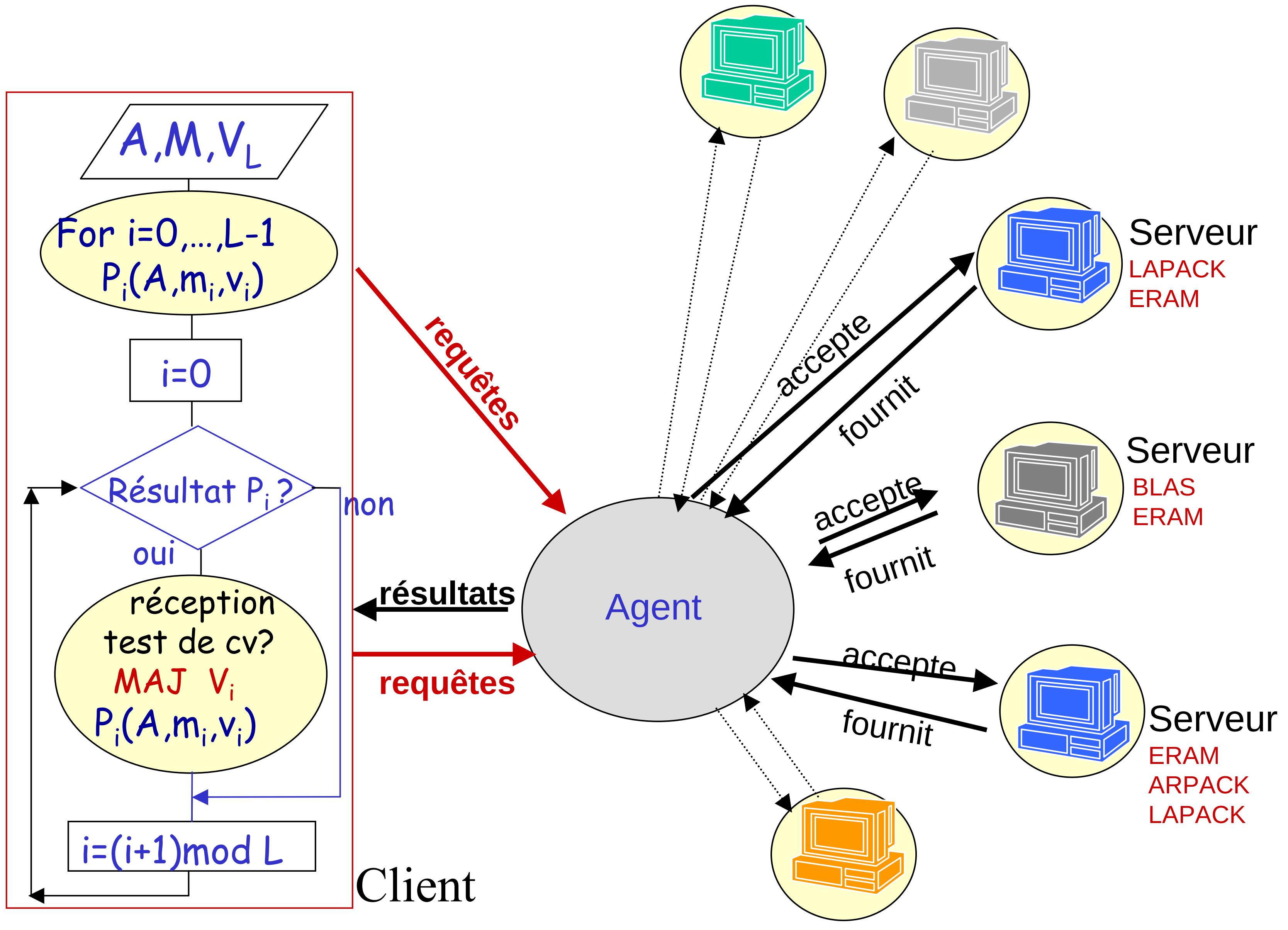
S : *Subspace computation*

Re : *Reduction of the results of 2 PRR processes*

R : *Restarting strategy*



PPR on NetSolve



omniRPC and YML

YML framework provides a tools to support the execution and the programming of the complex applications

- Middleware independence (XtremWeb, omniRPC, …)
- Graph description language (yvetteML)
- Easy integration of some existing LA libraries

omniRPC middleware

- Grid Remote Procedure Call System
- Master-worker style parallel programming
- Globus, ssh : Certification
- Persistency : Enable to store data on remote hosts
- Asynchronous call
- Automatic process management

Multiple PRR – YvetteML code

```
par (id := 1 ; nbProcess)
do # id is the PRR process identifier
  compute PRR_Start(I[id], n, id);
  seq (i := 1 ; maxIter)
  do
    compute PRR_Reduction(H[id], B[id],n, m[id], I[id], id);
    compute PRRSolver(Val[intNodes+id],Vec[intNodes + id],
            Res[intNodes + id],H[id], B[id], n, m[id], r, tol, id);
    # Reduction
    # restart
  enddo
enddo
```

omniRPC pseudo code

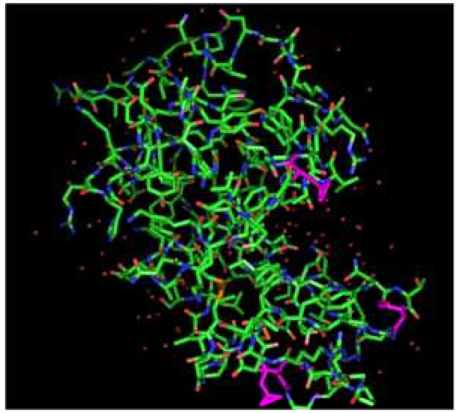
```
call OmniRPC_Init
call OmniRPC_Module_Init(. . .)
. . .
. . .
do j = 0, N-1
   . . .
   . . .
   call OmniRPC_Call_Async(. . .)
   . . .
   . . .
enddo
call OmniRPC_Wait_All
. . .
```

Specify initial data to send

Do not need to specify servers

Numerical Examples

A problem derived from computation of the molecular orbital of Lysozyme



A, B : Real symmetric
n = 20,758

Drop small matrix elements ($\leq 10^{-7}$):
Number of nonzero elements = 10,010,416

Parameters

N = 48

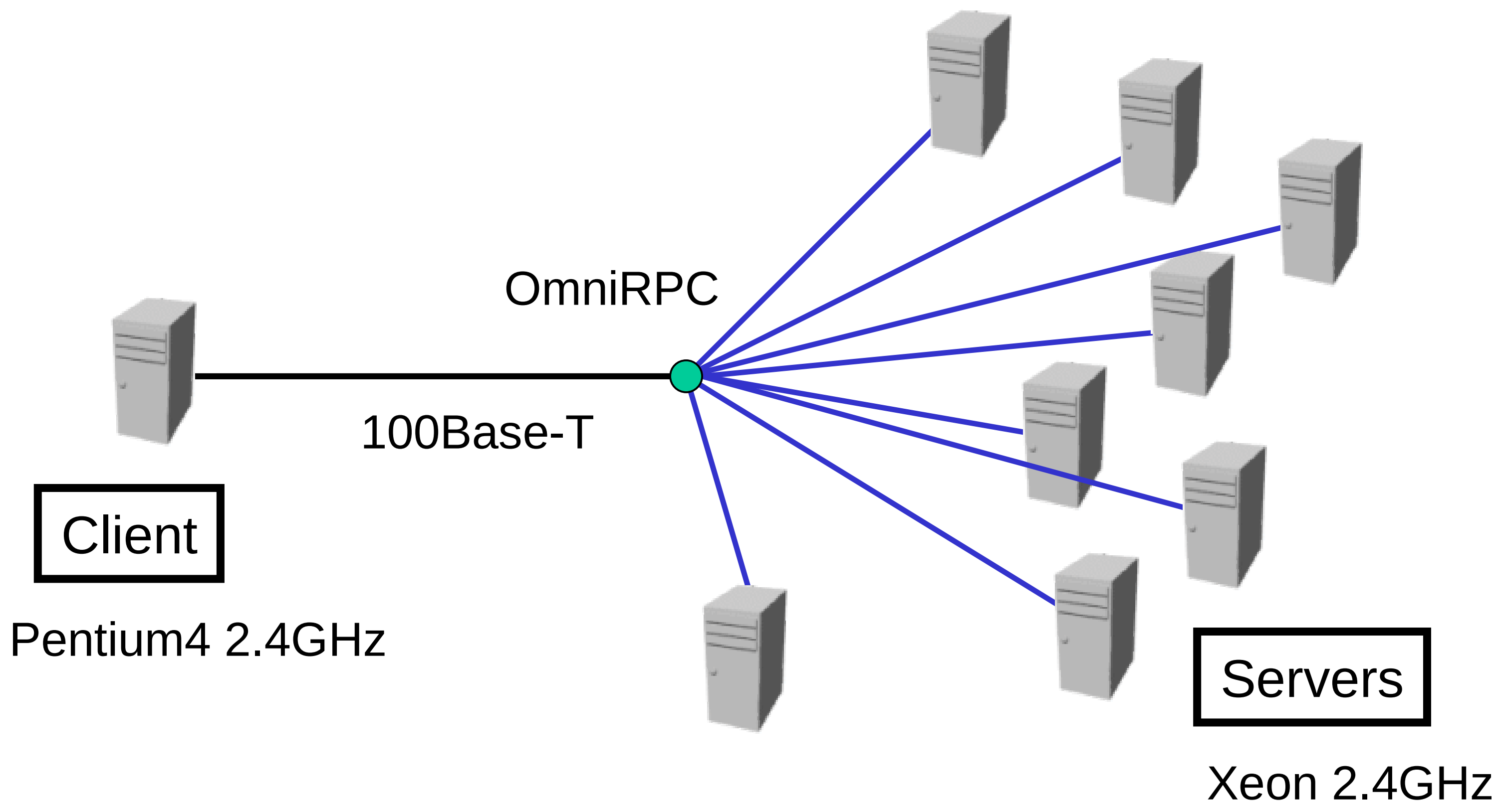
8 circles were located around the frontier orbital

20 eigenvalues were obtained

Solver for linear systems : COCG method

Preconditioner : incomplete Cholesky factorization

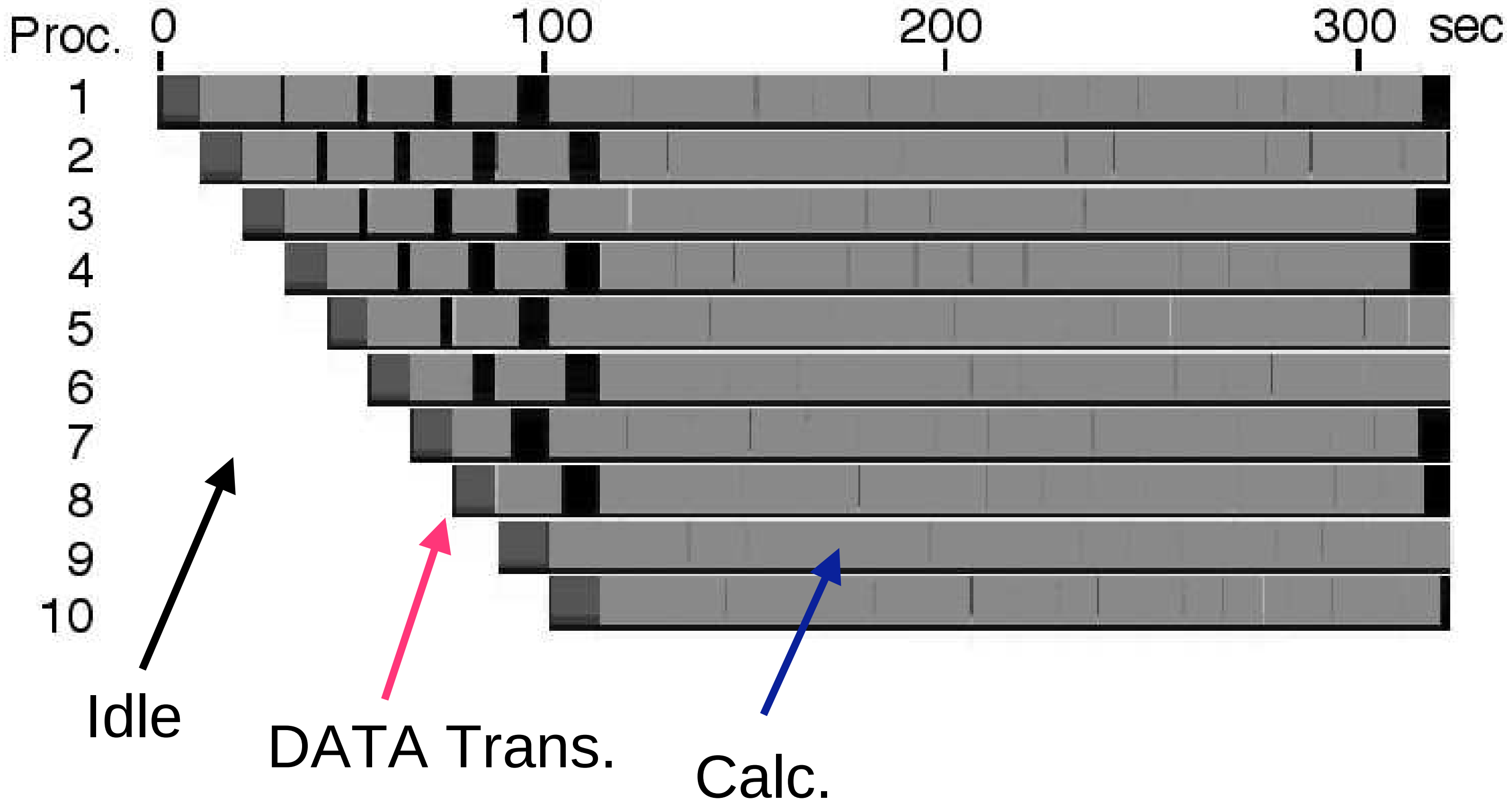
Runtime execution support



Wall-clock time in seconds

Number of nodes	Time (sec)	Speedup
1	2542	1.00
2	1271	2.00
4	665	3.82
6	466	5.45
8	375	6.78
10	318	7.99

Timing Results



Conclusion

- Sakurai-Sugiura and PRR methods need many computations (Coarse grain, Asynchronism, Fault tolerance, ...)
- Rehabilitation of this kind of methods
- Data transfer method (data warehouse solution)
- Performance evaluation criterion (execution time, ...)